Pautas de expresiones regulares y consideraciones de rendimiento para el filtrado de URL

Contenido

- Introducción
- Prerequisites
  - Requirements
  - Componentes Utilizados
- Antecedentes
  - Puntos clave
- Patrones que se deben evitar
- Prácticas recomendadas
  - Escapar siempre de los puntos en los hostnames
  - Patrones de anclaje y caracteres restringidos
  - Evite la repetición anidada y sin límites donde sea posible
  - Patrones de prueba en un probador compatible con PCRE2
- Diferencias en la coincidencia de URL para HTTP y HTTPS
  - Tráfico HTTPS (TLS)
  - Tráfico HTTP (no cifrado)
  - Implicaciones de configuración
- Verificación
  - Habilitar registro de depuración
- Ejemplos de Configuración
  - Coincidencia basada en host
  - Coincidencia de ruta/host HTTP
- Información Relacionada

Introducción

Este documento describe las pautas y las consideraciones de rendimiento para utilizar expresiones regulares en el filtrado de URL con el motor UTD. El filtrado de URL en el motor UTD utiliza la biblioteca de expresiones regulares PCRE2.

Colaboración de Eugene Khabarov, Cisco Engineering.

Prerequisites

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

- Sintaxis de expresiones regulares (regex)
- Conceptos de filtrado de URL
- Configuración de Unified Threat Defence (UTD)
- Diferencias de protocolo HTTPS/HTTP

Componentes Utilizados

Este documento no tiene restricciones específicas en cuanto a versiones de software y de hardware.

La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en funcionamiento con una configuración verificada (predeterminada). Si tiene una red en vivo, asegúrese de entender el posible impacto de cualquier comando.

Antecedentes

Mientras que PCRE2 es poderoso, ciertas expresiones complejas o 'codiciosas' pueden causar un seguimiento excesivo y pueden alcanzar límites internos en el motor regex. Cuando esto ocurre, un patrón puede tardar demasiado tiempo en procesarse y, en última instancia, tratarse como "sin coincidencia".

Puntos clave

- PCRE2 aplica límites internos en los pasos de retroceso o el tiempo de coincidencia para proteger los recursos del sistema.
- Algunos patrones son sintácticamente válidos pero computacionalmente inseguros y pueden desencadenar un 'retroceso catastrófico'.
- Cuando se exceden estos límites, el motor de regex puede abortar el procesamiento y no devolver ninguna coincidencia, incluso si la URL coincide lógicamente con el patrón.

Patrones que se deben evitar

Evite construcciones regex que combinen:

- Cuantificadores anidados, por ejemplo: (...+)*, (.*)*, (.+)+, etc.
- Los comodines (.) se repiten en grandes partes de la cadena, especialmente cerca del final del patrón
- Puntos sin escape en nombres de dominio cuando se utilizan junto con la repetición

Por ejemplo, aquí el patrón es sintácticamente válido pero puede ser costoso de procesar:

```
^([a-zA-Z0-9-]+.)*portal.example.com$
```

Nota: En este caso, ([a-zA-Z0-9-]+.)* es un grupo con un cuantificador anidado (+ dentro *) más un comodín (.). En algunas entradas que no coinciden, el motor regex puede explorar un gran número de trayectorias de retroceso.

Prácticas recomendadas

Escapar siempre de los puntos en los hostnames

Utilice \.para hacer coincidir un punto literal, por ejemplo:

```
^([a-zA-Z0-9-]+\.)*portal\.example\.com$
```

Patrones de anclaje y caracteres restringidos

Utilice ^ y $ y restrinja el uso de los caracteres esperados (por ejemplo, [a-zA-Z0-9-] para las etiquetas de host) para reducir el retroceso.

Evite la repetición anidada y sin límites donde sea posible

Prefiere construcciones más simples en lugar de patrones complejos que tratan de cubrir todo en un regex. Considere varias entradas específicas en lugar de una expresión muy amplia.

Patrones de prueba en un probador compatible con PCRE2

Antes de la implementación, pruebe los patrones de expresiones regulares en un entorno compatible con PCRE2 y evite los patrones que provoquen un "retroceso catastrófico" o advertencias similares.

Nota: Si un patrón regex alcanza los límites internos del motor PCRE2, el motor de filtrado de URL puede tratarlo como "no coincidencia". En estos casos, la clasificación de URL vuelve a la categoría o reputación, no al resultado de la lista blanca/lista negra. Los límites exactos son específicos de la implementación y pueden cambiar entre versiones. Debe diseñar regex de forma conservadora.

Diferencias en la coincidencia de URL para HTTP y HTTPS

El motor UTD inspecciona las URL de forma diferente para el tráfico HTTPS y HTTP. Esto afecta al modo en que se deben diseñar las expresiones regulares para el filtrado de URL.

Tráfico HTTPS (TLS)

Para el tráfico HTTPS cifrado, el motor UTD no descifra la carga de forma predeterminada.

- El filtrado de URL utiliza la indicación de nombre de servidor (SNI) del saludo del cliente de seguridad de la capa de transporte (TLS).
- El patrón regex se aplica solamente al nombre de host SNI, por ejemplo: api.example.com

En este caso, un patrón basado en hostname se compara con la cadena de hostname api.example.com como:

```
^([a-zA-Z0-9-]+\.)*example\.com$
```

Tráfico HTTP (no cifrado)

Para el tráfico HTTP normal, el motor UTD puede ver la solicitud HTTP completa (línea de solicitud y encabezados).

Dependiendo de la implementación, la cadena dada al motor regex puede incluir:

- La dirección URL completa o la línea de solicitud (por ejemplo, GET /path?param=value HTTP/1.1) o
- El encabezado Host combinado con la ruta (por ejemplo, api.example.com/path)

Como resultado, la entrada regex para HTTP puede contener caracteres adicionales como /, ? y cadenas de consulta, no sólo el nombre de host simple.

Implicaciones de configuración

Un regex diseñado exclusivamente para nombres de host (por ejemplo, sólo api.example.com coincidente) puede coincidir correctamente con HTTPS (SNI), pero no puede coincidir con una solicitud HTTP que contenga una URL completa o una cadena de ruta de acceso de host.

Para filtrar el tráfico HTTP y HTTPS con el mismo patrón, debe:

- Diseñar patrones principalmente en torno a nombres de host
- Verifique el comportamiento frente a HTTP y HTTPS en los registros UTD

Verificación

Habilitar registro de depuración

Paso 1. Ejecute el comando debug utd engine standard url-filters level info para habilitar el registro de depuración.

Paso 2. Ejecute el comando show logging process ioxman module utd | include api.example.com para verificar los registros.

Ejemplo de salida:

```
2025/11/27 11:45:28.195000350 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF event->server_
2025/11/27 11:45:28.195001873 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF   URL: api.exa
2025/11/27 11:45:28.195009216 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF Regex matched
2025/11/27 11:45:28.195022442 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF URLF whitelist
2025/11/27 11:45:33.530605572 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF   URL: api.exa
2025/11/27 11:45:33.530606333 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF Regex not matc
2025/11/27 11:45:33.530614980 {ioxman_R0-0}{255}: [utd] [21292]: (note): :(#0):INSP-URLF URLF whitelist
```

Ejemplos de Configuración

Coincidencia basada en host

Para permitir todos los subdominios de example.com, utilice este patrón recomendado centrado en el nombre de host (línea de base):

```
^([a-zA-Z0-9-]+\.)*example\.com$
```

Este patrón:

- Coincide con example.com, api.example.com, foo.bar.example.com, etc
- Es adecuado para la coincidencia de HTTPS (SNI)
- También puede coincidir con HTTP si la cadena vista por el motor es el nombre de host sin software específico

Coincidencia de ruta/host HTTP

Si HTTP incluye host/ruta y desea ignorar la ruta, puede hacer coincidir el prefijo del nombre de host y dejar que regex se detenga en un límite de palabra en lugar de en un límite final. *, por ejemplo:

```
^([a-zA-Z0-9-]+\.)*example\.com\b
```

Nota: Aquí, \b (límite de palabra) permite caracteres tales como / o ? para seguir el nombre de host sin requerir un comodín .* explícito. Esto es generalmente más barato que añadir .* al final y se alinea mejor con la guía para evitar comodines sin límite adicionales.

Precaución: La cadena exacta que se pasa al motor regex para las solicitudes HTTP es

---

⚠ específica de la implementación y puede evolucionar. En caso de duda, pruebe los patrones con el tráfico HTTP y HTTPS en un entorno de laboratorio y verifique las coincidencias en los registros UTD antes de implementar en producción.

Información Relacionada

- Guía de Configuración de Seguridad de Cisco Catalyst SD-WAN, Cisco IOS XE Catalyst SD-WAN Release 17.x
- Soporte técnico y descargas de Cisco

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).